Numer Algor (2013) 63:601–614
DOI 10.1007/s11075-012-9644-0

ORIGINAL PAPER

Algorithm for min-range multiplication of affine forms

Iwona Skalna

Received: 8 November 2011 / Accepted: 30 August 2012 / Published online: 16 September 2012

Abstract Affine arithmetic produces guaranteed enclosures for computed quantities, taking into account any uncertainties in the input data as well as round-off errors. Elementary operations on affine forms are redefined so they result in affine forms. Affine-linear operations result straightforwardly in affine forms. Non-linear operators, such as multiplication, must be approximated by affine forms. Choosing the appropriate approximation is a big challenge. The reason is that different approximations may be more accurate for specific purposes. This paper presents an efficient method for computing the minimum range (min-range) affine approximation of the product of arbitrary affine forms that do not contain zero properly. Numerical experiments are carried out to demonstrate the essential features of the proposed approach, especially its usefulness for bounding ranges of functions for global optimisation and for finding roots of functions.

Keywords Affine arithmetic · Multiplication · Min-range approximation · Range bounding · Global optimisation · Roots of functions

Notation

$\hat{x}$	an affine form
$\langle \hat{x}, \hat{y} \rangle$	the joint range of two affine forms $\hat{x}$ and $\hat{y}$
$\boldsymbol{x}$, $[x]$	a closed interval
$\underline{x}$	the lower bound (left endpoint) of an interval $\boldsymbol{x}$
$\overline{x}$	the upper bound (right endpoint) of an interval $\boldsymbol{x}$
$\mathrm{int}(\boldsymbol{x})$	the interior of an interval $\boldsymbol{x}$

I. Skalna (✉)
AGH University of Science and Technology, Krakow, Poland
e-mail: skalna@agh.edu.pl

$\check{x} = (\overline{x} + \underline{x})/2$	the midpoint of an interval $\boldsymbol{x}$
$\mathrm{r}(\boldsymbol{x}) = (\overline{x} - \underline{x})/2$	the radius of an interval $\boldsymbol{x}$
$[a, b]$	an interval with lower bound a and upper bound b
$\hat{\boldsymbol{x}}$, $[\hat{x}]$	the interval range of (interval corresponding to) an affine form $\hat{x}$
$\boldsymbol{x} >= 0$ ($\boldsymbol{x} <= 0$)	an interval of nonnegative (nonpositive) real numbers
∂S	the border of a set S

1 Introduction

Affine arithmetic (AA) was first introduced by Comba and Stolfi in 1993 [2] as a new self-validated model for numerical computation. It was designed to eliminate the main weakness of standard interval arithmetic (IA) [11], that is the tendency to produce intervals which are often much wider than the true range of the corresponding quantities, especially in long computation chains. AA is similar to standard interval arithmetic in that it keeps track of input, truncation, and rounding errors. In addition, it takes into account correlations between computed and input quantities, and is, therefore, able to provide much tighter bounds on computed quantities than standard interval arithmetic.

In affine arithmetic [2, 4–6] an unknown ideal quantity x is represented by an affine form $\hat{x} = x_0 + x_1\varepsilon_1 + \ldots + x_n\varepsilon_n$ which is a degree 1 polynomial. The central value x_0 and the *partial deviations* x_i are finite floating-point numbers; the *noise symbols* ε_i are unknown but assumed to vary within their domains $\boldsymbol{D}_i = [-1, 1] \ni \varepsilon_i$. Affine forms sharing the same noise symbols are partially correlated through them [3]. All possible pairs $(\hat{x}, \hat{y})$ (assuming that each ε_i vary independently within $\boldsymbol{D}_i$) lie in a convex polygon (zonotope) which is called a *joint range* and is denoted by $\langle \hat{x}, \hat{y} \rangle$ ($\langle \hat{x}, \hat{x} \rangle$ reduces to a line).

Every affine form $\hat{x}$ implies the range $[\hat{x}] = [x_0 - \mathrm{r}_x, x_0 + \mathrm{r}_x]$ for an unknown ideal quantity x, which is the smallest interval that contains all possible values of $\hat{x}$, assuming that each ε_i varies independently within the domain $\boldsymbol{D}_i$. The radius $\mathrm{r}_x = \sum_{i=1}^{n} |x_i|$ is called the *total deviation* of $\hat{x}$ [5]. Conversely, if an ideal quantity x belongs to an interval $\boldsymbol{x} = [\underline{x}, \overline{x}]$, then x can be represented by an affine form $\hat{x} = \check{x} + \mathrm{r}(\boldsymbol{x})\varepsilon_i$, where $\check{x} = (\underline{x} + \overline{x})/2$ is a midpoint of $\boldsymbol{x}$, $\mathrm{r}(\boldsymbol{x}) = (\overline{x} + \underline{x})/2$ is a radius of $\boldsymbol{x}$, and ε_i is a noise symbol not occurring in any previous computations [5].

Affine-linear operations on affine forms result straightforwardly in affine forms. Non-affine operations must be approximated by affine forms. An extra term must then be added to bound the error of this approximation (this extra term usually also includes round-off errors). Selecting appropriate affine approximation might reduce this error. Given a non-affine function of two variables $z = f(x, y)$ and two affine forms $\hat{x}$ and $\hat{y}$ representing x and y, an affine form $\hat{z}$ representing z must be computed. It is desirable that $\hat{z}$ is consistent with $\hat{x}$ and $\hat{y}$ and that it preserves the information provided by them

as much as possible [6]. It can be easily seen that $z = f(\hat{x},\ \hat{y})$ is a function of the noise symbols ε_i:

$$\begin{aligned} z &= f(x_0 + x_1\varepsilon_1 + \ldots + x_n\varepsilon_n,\ y_0 + y_1\varepsilon_1 + \ldots + y_n\varepsilon_n) \\ &= f^*(\varepsilon_1,\ \ldots,\ \varepsilon_n), \end{aligned} \tag{1}$$

where $f^* : \boldsymbol{D}_1 \times \ldots \times \boldsymbol{D}_n \longrightarrow \mathbb{R}$ is generally a non-affine function. An affine approximation f^a of f^* can be then written in the form:

$$f^a = z_0 + z_1\varepsilon_1 + \ldots + z_n\varepsilon_n + z_k\varepsilon_k, \tag{2}$$

where the last term $z_k\varepsilon_k$ represents the *residual* or *approximation error*. It is assumed that ε_k is a new noise symbol independent from $\varepsilon_1,\ \ldots,\ \varepsilon_n$ [4].

The quality of the approximation depends on the selection of a central value z_0 and partial deviations z_i. This means that there are $n + 1$ degrees of freedom for the choice of an affine approximation f^a. In fact, two basic approaches to compute affine approximation are used the most frequently [6]. The first one is to minimise the approximation error z_k (Chebyshev approximation), the second one is to minimise the range $[\hat{x} \cdot \hat{y}]$ (minimum range or shortly min-range approximation). The choice of which affine approximation to use depends on the problem to be solved. In some applications it is important to compute interval bounds which contain only positive numbers, e.g. computation of the square root. In such cases, the min-range approximation should be chosen. The range optimality is also needed in computer graphics (cf. [3, 9, 10]) or when the denominator of an expression is an affine form. In the latter case, the narrower interval is less likely to contain zero.

Let us now consider the multiplication of affine forms, that is, the evaluation of $z = f(x,\ y) = xy$, given the affine forms $\hat{x}$ and $\hat{y}$ for x and y. Since the product of affine forms is a quadratic polynomial on the noise symbols, it must, therefore, be approximated by an affine form. In [7], a formula for the product of affine forms that do not contain zero properly was suggested. It yields no overestimation (i.e. gives a minimal range) if certain simple monotonicity conditions are valid. Otherwise, only a suboptimal range can be obtained. The monotonicity conditions are required to calculate global extrema of the product on a joint range $\langle \hat{x},\ \hat{y} \rangle$.

This work generalises the result from [7] by relaxing monotonicity requirements. In Section 3 the algorithm for computing the required global extrema is proposed. Then, the minimal range product can be obtained for arbitrary affine forms that do not contain zero properly. The rest of the paper is organised as follows. The theoretical framework of the paper is presented in Section 2 and Section 3. Numerical experiments illustrating the features of the proposed method are provided in Section 4. The paper ends with concluding remarks.

2 Multiplication of affine forms

In this section, the formula for optimal multiplication of affine forms (in the min-range sense) is introduced. As it was mentioned above, only affine forms that do not contain zero properly ($0 \notin \mathrm{int}([\hat{x}])$) are considered.

Remark If at least one of the affine forms $\hat{x}$, $\hat{y}$ contains zero properly, another non-optimal multiplication formula must be used. One can exploit the trivial multiplication (TR) described in [4] or the multiplication formula introduced in [7, 8]. Throughout this paper, the latter will be called standard multiplication (ST).

The following four cases can be distinguished:

(1) $[\hat{x}] \geqslant 0, [\hat{y}] \geqslant 0$,
(2) $[\hat{x}] \geqslant 0, [\hat{y}] \leqslant 0$,
(3) $[\hat{x}] \leqslant 0, [\hat{y}] \geqslant 0$,
(4) $[\hat{x}] \leqslant 0, [\hat{y}] \leqslant 0$.

In fact, it is enough to consider three of the four cases, given that the second and third are symmetrical. Moreover, the cases (2) and (4) can easily be reduced to the case (1). For example, in the case (2), where $[\hat{x}] \geqslant 0$ and $[\hat{y}] \leqslant 0$, the product

$$\hat{z} = \hat{x} \cdot \hat{y} = -\hat{x}(-\hat{y}) = -\hat{x}\hat{y}' = -\hat{z}',$$

where $\hat{z}'$ is a product of non-negative affine forms $\hat{x}$ and $\hat{y}'$. Thus, in what follows, the focus is on the first case.

Following [7], the product $\hat{z}$ of $\hat{x}$, $\hat{y}$ ($[\hat{x}] = [\underline{x}, \overline{x}] \geqslant 0$, $[\hat{y}] = [\underline{y}, \overline{y}] \geqslant 0$) of the form

$$\hat{z} = -\underline{x}\,\underline{y} + \underline{y}\hat{x} + \underline{x}\hat{y} + \boldsymbol{b}, \tag{3}$$

is searched. Here $\boldsymbol{b}$ is an unknown interval and the goal is to determine $\boldsymbol{b}$ such that $\hat{z}$ implies the minimal range $\boldsymbol{z}^* = [\underline{z}^*, \overline{z}^*]$, where

$$\underline{z}^* = \min_{D_1 \times \ldots \times D_n} f^*(\varepsilon_1, \ldots, \varepsilon_n),$$

$$\overline{z}^* = \max_{D_1 \times \ldots \times D_n} f^*(\varepsilon_1, \ldots, \varepsilon_n).$$

In the next section, an efficient method for computing $\boldsymbol{z}^*$ is proposed. The method does not require any monotonicity conditions to be fulfilled and, therefore, allows to considerably generalise the result from [7].

3 Main result

The joint range of two affine forms with n noise symbols has $2n$ parallel sides and is symmetric around the central point $(x_0,\ y_0)$. Each ε_i corresponds to a pair of opposite sides, which are parallel and congruent to the segment with endpoints $(x_i,\ y_i)$ and $(-x_i,\ -y_i)$ [4].

According to (1) with $f(x,\ y) = xy$, extrema of f^* on $\boldsymbol{D}_1 \times \ldots \times \boldsymbol{D}_n$ are equal to extrema of f on the joint range $\langle \hat{x},\ \hat{y} \rangle$.

Theorem 1 *Let x and y be real values consistent with affine forms $\hat{x}$ and $\hat{y}$. Then, the function*

$$f(x,\ y) = x \cdot y$$

attains its extrema on the border $D = \partial\langle \hat{x},\ \hat{y} \rangle$ of the joint range $\langle \hat{x},\ \hat{y} \rangle$.

Proof The partial derivatives $f_x(x,\ y) = y$ and $f_y(x,\ y) = x$ are zero only at $(0,\ 0)$. Since $f_{xx} f_{yy} - f_{xy}^2 < 0$, the critical point is a saddle point and, therefore, the extrema have to be at the boundary. □

Theorem 1 states that, in order to calculate $\underline{z}^*$ and $\overline{z}^*$ it is enough to consider the border of the joint range $\langle \hat{x},\ \hat{y} \rangle$. As mentioned before, the joint range of two affine forms of length n has $2n$ parallel sides. Therefore, the problem reduces to computing minimal and maximal values on each of $2n$ sides.

Let us now consider the system of equations

$$\begin{cases} x = x_0 + x_1\varepsilon_1 + \ldots + x_n\varepsilon_n, \\ y = y_0 + y_1\varepsilon_1 + \ldots + y_n\varepsilon_n, \end{cases} \tag{4}$$

and let arbitrary i be chosen. Eliminating ε_i from (4) gives

$$x_i y - y_i x = x_i y_0 - y_i x_0 + \sum_{\substack{j=1 \\ i \neq j}}^{n} (x_i y_j - y_i x_j)\varepsilon_j, \tag{5}$$

which defines a line parallel to the sides corresponding to the ε_i.

If $x_i \neq 0$, then to reach one of those sides, ε_j $(j \neq i)$ should be selected such that the y-intercept is minimal. To reach the other (parallel) side, the ε_j $(j \neq i)$ should be selected such that the y-intercept is maximal. This suggests the following selection procedure for computing the minimal value of y-intercept:

$$\varepsilon_j = \begin{cases} +1, & y_j - (y_i/x_i)x_j \leq 0, \\ -1, & \text{otherwise.} \end{cases} \tag{6}$$

If $x_i = 0$, (5) takes the form

$$x = x_0 + \sum_{\substack{j=1 \\ i \neq j}}^{n} x_j \varepsilon_j, \tag{7}$$

and (6) should be replaced by

$$\varepsilon_j = \begin{cases} +1, & x_j \le 0, \\ -1, & \text{otherwise.} \end{cases} \tag{8}$$

Similar procedure can be applied to calculate maximal y-intercept.

Assuming that all ε_j $(j \neq i)$ are fixed to $+1$ or -1 according to (6) or (8), i.e. the respective side is reached, the function f^* becomes a second degree polynomial in ε_i:

$$f^* = f^*(\varepsilon_i) = a\varepsilon_i^2 + b\,\varepsilon_i + c, \tag{9}$$

where

$$\begin{aligned} a &= x_i y_i, \\ b &= (x_0 + a_x) y_i + (y_0 + a_y) x_i, \\ c &= (x_0 + a_x)(y_0 + a_y), \end{aligned} \tag{10}$$

and

$$a_x = \sum_{\substack{j=1 \\ i \neq j}}^{n} x_j \varepsilon_j\,, \quad a_y = \sum_{\substack{j=1 \\ i \neq j}}^{n} y_j \varepsilon_j. \tag{11}$$

Obviously, the maximum $\overline{m}_i$ and minimum $\underline{m}_i$ of f^* in the range $[-1,\ 1]$ is, respectively, the maximum and minimum of f^* on that side. Finally,

$$\begin{aligned} \underline{z}^* &= \min\{\underline{m}_i \mid i = 1,\ \ldots,\ 2n\}, \\ \overline{z}^* &= \min\{\overline{m}_i \mid i = 1,\ \ldots,\ 2n\}. \end{aligned}$$

Once $\underline{z}^*$ and $\overline{z}^*$ are computed, the endpoints of interval $\boldsymbol{b}$ in (3) can be determined in the following way:

$$\begin{aligned} \underline{b} &= \underline{z}^* + \underline{x}\,\underline{y} - \underline{y} x_0 - \underline{x} y_0 + \sum_{i=1}^{n} |\underline{y} x_i + \underline{x} y_i|, \\ \overline{b} &= \overline{z}^* + \underline{x}\,\underline{y} - \underline{y} x_0 - \underline{x} y_0 - \sum_{i=1}^{n} |\underline{y} x_i + \underline{x} y_i|. \end{aligned}$$

Finally, (3) can be written in a form:

$$\hat{z} = z_0 + \sum_{i=1}^{n} z_i \varepsilon_i + z_k \varepsilon_k, \tag{12}$$

where

$$\begin{aligned} z_0 &= -\underline{x}\,\underline{y} + \underline{y} x_0 + \underline{x} y_0 + (\underline{b} + \overline{b})/2 = -\underline{x}\,\underline{y} + \underline{y} x_0 + \underline{x} y_0 + \check{b}, \\ z_i &= \underline{y} x_i + \underline{x} y_i, \\ z_k &= (\overline{b} - \underline{b})/2 = \mathrm{r}(\boldsymbol{b}). \end{aligned} \tag{13}$$

Theorem 2 *Let $\hat{x}$ and $\hat{y}$ be arbitrary non-negative affine forms of length n and let $[\hat{x}] = [\underline{x},\ \overline{x}] \geqslant 0$ and $[\hat{y}] = [\underline{y},\ \overline{y}] \geqslant 0$. Then, the product*

$$\hat{z} = z_0 + z_1\varepsilon_1 + \ldots + z_n\varepsilon_n + z_k\varepsilon_k,$$

of $\hat{x}$ and $\hat{y}$ reduces to minimal range $\boldsymbol{z}^*$ if the noise symbols z_i $(i = 1, \ldots, n)$ and the approximation error z_k are determined as follows:

$$\begin{aligned} z_0 &= (\underline{z}^* + \overline{z}^*)/2, \\ z_i &= \underline{y}x_i + \overline{x}y_i, \quad i = 1, \ldots, n, \\ z_k &= (\overline{z}^* - \underline{z}^*)/2 - R, \\ R &= \sum_{i=1}^{n} |z_i|. \end{aligned} \tag{14}$$

Proof According to (3),

$$\hat{z} = -\underline{x}\underline{y} + \underline{y}\hat{x} + \underline{x}\hat{y} + \boldsymbol{b} = z_0 + \sum_{i=1}^{n} z_i + z_k\varepsilon_k, \tag{15}$$

where

$$\begin{aligned} z_0 &= z_0' + \check{b}, \quad z_0' = -\underline{x}\underline{y} + \underline{x}y_0 + \underline{y}x_0, \\ z_i &= \underline{y}x_i + \underline{x}y_i, \\ z_k &= \mathrm{r}(\boldsymbol{b}). \end{aligned} \tag{16}$$

Then, the range of $\hat{z}$

$$\begin{aligned} [\hat{z}] &= \left[z_0' + \check{b} - \sum_{i=1}^{n} |z_i| - \mathrm{r}(\boldsymbol{b}),\ z_0' + \check{b} + \sum_{i=1}^{n} |z_i| + \mathrm{r}(\boldsymbol{b}) \right] \\ &= \left[z_0' - \sum_{i=1}^{n} |z_i| + \underline{b},\ z_0' + \sum_{i=1}^{n} |z_i| + \overline{b} \right]. \end{aligned}$$

Since it is desired that $[\hat{z}] = [\underline{z}^*,\ \overline{z}^*]$, thus the following equalities must hold:

$$\underline{z}^* = z_0' - \sum_{i=1}^{n} |z_i| + \underline{b}, \quad \overline{z}^* = z_0' + \sum_{i=1}^{n} |z_i| + \overline{b}.$$

This implies

$$\begin{aligned} \underline{b} &= \underline{z}^* - z_0' + \sum_{i=1}^{n} |z_i| = \underline{z}^* - z_0' + R, \\ \overline{b} &= \overline{z}^* - z_0' - \sum_{i=1}^{n} |z_i| = \overline{z}^* - z_0' - R. \end{aligned}$$

Then,

$$\check{b} = (\underline{z}^* + \overline{z}^*)/2 - z_0',$$
$$\mathrm{r}(\boldsymbol{b}) = (\underline{z}^* + \overline{z}^*)/2 - R, \tag{17}$$

Substituting $\check{b}$ and $\mathrm{r}(\boldsymbol{b})$ into (16) gives (14). □

The outline of the algorithm for computing the extremal values $\underline{z}^*$ and $\overline{z}^*$ is presented in Algorithm 1. In the outer for-loop ($i = 1, \ldots, n$), the border of the joint range is searched for extreme values of the product of affine forms.

Algorithm 1 Computing the optimal range: $[\underline{z}^*,\ \overline{z}^*]$

Input: $\hat{x} = x_0 + \sum_{i=1}^{n} x_i\varepsilon_i,\ \hat{y} = y_0 + \sum_{i=1}^{n} y_i\varepsilon_i$

Output: $\boldsymbol{z} = [\underline{z}^*,\ \overline{z}^*]$ $(\underline{z}^* = \min_{\langle \hat{x},\, \hat{y} \rangle} xy,\ \overline{z}^* = \max_{\langle \hat{x},\, \hat{y} \rangle} xy)$

$\underline{z}^* = +\infty;\ \overline{z}^* = -\infty;$
for $i = 1$ **to** n **do**
  $\underline{a}_x = \overline{a}_x = \underline{a}_y = \overline{a}_y = 0;$
  for $j = 1$ **to** n **do**
    if $(j \neq i)$ **then**
      if $(x_i = 0)$ **then**
        if $(x_j \geqslant 0)$ **then** $\underline{\varepsilon}_j = 1;\ \overline{\varepsilon}_j = -1;$ **else** $\underline{\varepsilon}_j = -1;\ \overline{\varepsilon}_j = 1;$
      else if $(y_i = 0)$ **then**
        if $(y_j \geqslant 0)$ **then** $\underline{\varepsilon}_j = 1;\ \overline{\varepsilon}_j = -1;$ **else** $\underline{\varepsilon}_j = -1;\ \overline{\varepsilon}_j = 1;$
      else
        if $(-y_i x_j / x_i + y_j \geqslant 0)$ **then** $\underline{\varepsilon}_j = 1;\ \overline{\varepsilon}_j = -1;$ **else** $\underline{\varepsilon}_j = -1;\ \overline{\varepsilon}_j = 1;$
      end if
    end if
    $\underline{a}_x = \underline{a}_x + x_j\underline{\varepsilon}_j;\ \overline{a}_x = \overline{a}_x + x_j\overline{\varepsilon}_j;$
    $\underline{a}_y = \underline{a}_y + y_j\underline{\varepsilon}_j;\ \overline{a}_y = \overline{a}_y + y_j\overline{\varepsilon}_j;$
  end for
  $\underline{a} = x_i y_i;\ \underline{b} = x_0 y_+ y_0 x_i + \underline{a}_x y_i + \underline{a}_y x_i;\ \underline{c} = \underline{a}_x\underline{a}_y + x_0 y_0 + x_0\underline{a}_y + y_0\underline{a}_x;$
  $\overline{a} = \underline{a};\ \overline{b} = x_0 y_+ y_0 x_i + \overline{a}_x y_i + \overline{a}_y x_i;\ \overline{c} = \overline{a}_x\overline{a}_y + x_0 y_0 + x_0\overline{a}_y + y_0\overline{a}_x;$
  $\underline{m} = \min\{\underline{a}\varepsilon_i^2 + \underline{b}\varepsilon_i + \underline{c}\ ,\ \varepsilon_i \in [-1,\ 1]\};\ \overline{m} = \max\{\underline{a}\varepsilon_i^2 + \underline{b}\varepsilon_i + \underline{c}\ ,$
    $\varepsilon_i \in [-1,\ 1]\};$
  $\underline{z}^* = \min\{\underline{z}^*,\ \underline{m}\};\ \overline{z}^* = \max\{\overline{z}^*,\ \overline{m}\};$
  $\underline{m} = \min\{\overline{a}\varepsilon_i^2 + \overline{b}\varepsilon_i + \overline{c}\ ,\ \varepsilon_i \in [-1,\ 1]\};\ \overline{m} = \max\{\overline{a}\varepsilon_i^2 + \overline{b}\varepsilon_i + \overline{c}\ ,$
    $\varepsilon_i \in [-1,\ 1]\};$
  $\underline{z}^* = \min\{\underline{z}^*,\ \underline{m}\};\ \overline{z}^* = \max\{\overline{z}^*,\ \overline{m}\};$
end for
return $\boldsymbol{z} = [\underline{z}^*,\ \overline{z}^*];$

The border consists of n pairs of parallel sides. They are searched simultaneously, therefore, the loop performs n times. Sometimes, sides corresponding to different noise symbols coincide. The coinciding sides can be treated as one and this way the number of loops can be reduced.

The values corresponding to one side are underscored and those corresponding to the other side are overscored. In the inner for-loop ($j = 1, \ldots, n$), the noise symbols $\underline{\varepsilon}_j$ and $\overline{\varepsilon}_j$ are set to $+1$ or -1 according to formula (8) or (6) and the respective sums $\underline{a}_x, \underline{a}_y$ and $\overline{a}_x, \overline{a}_y$ (see formula (11)) are recalculated. After the inner loop is finished, the latter are used to compute the coefficients $\underline{a}, \underline{b}, \underline{c}$ and $\overline{a}, \overline{b}, \overline{c}$ of the respective second degree polynomials (9). Finally, the extremal values on each of the two parallel sides are obtained and the previously found extrema are updated.

The time complexity of the proposed algorithm is $O(n^2)$ in the number of noise symbols, which is one order of magnitude greater then that of ST algorithm [7]. In order to verify the practical use of the developed method, numerical experiments are performed in the next section.

4 Numerical experiments

This section presents various experiments aiming at showing the essential features of the new algorithm for min-range multiplication. The main interest is focused on the computational efficiency of the MR multiplication, since the optimality of the MR range results was proved in Theorem 2. The standard multiplication (ST) is used in this paper for comparison purposes. The advantages of the MR multiplication over the ST one are underlined. All calculations were performed on Intel Core i5 under Windows 7×64 with 3GB memory using the code developed by the author in C++.

Example 1 (Multiplication of random affine forms) In this example, 200,000 pairs of affine forms corresponding to the cases (1)–(4) are generated randomly. Each random affine form is defined by a central value and noise symbols which are in turn pseudo-random numbers from the uniform distribution on the following intervals: $[-10.0, 10.0]$ and $[-3.5, 3.5]$, $[-10, 10]$ and $[-4.5, 4.5]$, $[-20.0, 20.0]$ and $[-4.5, 4.5]$, $[-20, 20]$ and $[-5.5, 5.5]$. The first interval of each pair corresponds to the central value, the second one to the noise symbols. The number of noise symbols varies from 4 to 8.

The multiplication results are summarised in Table 1. Firstly, the accuracy of the bounds is compared. When both multiplicands have the same sign, then the resulting range should have the nonnegative lower bound, while the upper bound should be nonpositive when the multiplicands have opposite signs. In order to see whether it holds, the average of lower bounds is given in columns 2 and 4, and the average of upper bounds is given in columns 3, and 5. Additionally, the percentage of cases where ST gives too wide ranges, which means the min-range formula is needed, is given in the sixth column.

Table 1 The percentage (P) of cases where the ranges resulting from standard multiplication contain zero and the ratio of ST CPU time to MR CPU time per 200,000 multiplications

Variant	Case				P	$\frac{\text{MR CPU time}}{\text{ST CPU time}}$
	(1)	(2)	(3)	(4)		
[−10, 10], [−3.5, 3.5]	−53.525	50.918	−48.820	50.895	95 %	1.10
	8.412	−7.726	6.986	−7.679		
[−10, 10], [−4.5, 4.5]	−61.042	57.897	−54.172	57.705	96 %	1.04
	7.404	−6.834	6.344	−6.850		
[−20, 20], [−4.5, 4.5]	−125.171	123.177	−121.784	122.472	88 %	1.24
	41.913	−39.693	37.484	−39.705		
[−20, 20], [−5.5, 5.5]	−160.134	156.238	−152.759	155.457	92 %	1.15
	37.028	−35.191	32.955	−35.070		

The results show that the average of lower bounds is significantly lower, respectively, the average of upper bounds is significantly greater than that of MR. Besides, as depicted in Fig. 1, in over 40 % cases the standard multiplication ranges are at least one and half times wider than the min-range results.

What is more, in an average of 90 % of cases, the standard multiplication produced the ranges with the lower or upper bound having the sign opposite to the expected one. It is worth to underline that this feature of multiplication procedure is not welcome, especially when dealing with algebraic expressions involving square root, logarithmic or rational functions. Therefore, the min-range multiplication seems to be a good alternative for standard (or trivial) multiplication when computing such functions.

On the other hand, the CPU time speaks in favour of the standard multiplication method. The ratio of timings ($\frac{\text{MR CPU time}}{\text{ST CPU time}}$), given in the seventh column of Table 1, varies from 1.04–1.24. However, as it will be shown later (see Example 3), the longer time required to calculate the product is often compensated by narrower bounds, and this in turn can shorten the time of more complex calculations.

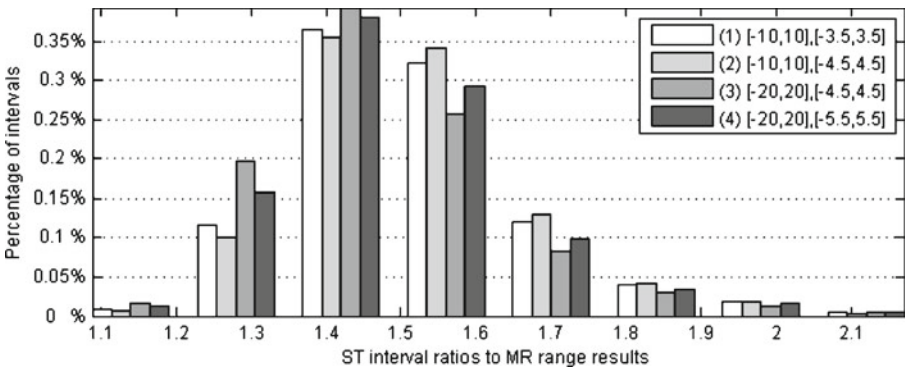

Fig. 1 Standard multiplication interval ratios to min-range range results

Example 2 (A square root problem) Let the following function involving a square root be given:

$$f(x,\ y) = \sqrt{0.26(x^2 + y^2) + 0.48xy}$$

and let $x,\ y \in [0.5,\ 3.5]$. Transforming x and y to affine forms gives:

$$\hat{x} = 2 + 1.5\varepsilon_1,$$
$$\hat{y} = 2 + 1.5\varepsilon_2.$$

It is worth noting that two different noise symbols, ε_1 and ε_2, are used to ensure that x and y vary independently within their intervals.

The range of f calculated with the use of min-range approximation equals $[0.5,\ 3.5]$ which is the optimal range for f. It is easy to verify that:

$$\min_{\varepsilon_1,\,\varepsilon_2 \in [-1,\,1]} \sqrt{0.26((2 + 1.5\varepsilon_1)^2 + (2 + 1.5\varepsilon_2)^2) + 0.48(2 + 1.5\varepsilon_1)(2 + 1.5\varepsilon_2)} = 0.5$$

$$\max_{\varepsilon_1,\,\varepsilon_2 \in [-1,\,1]} \sqrt{0.26((2 + 1.5\varepsilon_1)^2 + (2 + 1.5\varepsilon_2)^2) + 0.48(2 + 1.5\varepsilon_1)(2 + 1.5\varepsilon_2)} = 3.5$$

Using trivial and standard multiplication formula the partially negative ranges for the argument of the square root are obtained:

$$[0.26((2 + 1.5\varepsilon_1)^2 + (2 + 1.5\varepsilon_2)^2]_{\mathrm{TR}} = [-4.25,\ 12.25]$$

and

$$[0.26((2 + 1.5\varepsilon_1)^2 + (2 + 1.5\varepsilon_2)^2]_{\mathrm{ST}} = [-3.08,\ 12.25].$$

Then, the square root cannot be calculated in reals. Thus, in this case, only min-range multiplication was able to produce the result which can be further processed.

Example 3 (Global optimisation) To illustrate the effectiveness of inclusion functions based on the min-range multiplication formula, ten polynomial global minimisation benchmark problems (see, e.g., [12–14]) are presented and solved by using an interval branch-and-bound (B&B) based algorithm. The computational accuracy of the results in most of cases is set to 10^{-12} ($\epsilon = 10^{-12}$). The algorithm terminates when $|\tilde{f} - \underline{f}| \leqslant \epsilon$, where $\tilde{f}$ is the current midpoint value and $\underline{f} = \underline{f(\boldsymbol{x})}$ with $\boldsymbol{x}$ being the box that is currently processed. The functions, initial search domains and global minima are summarised in Table 2.

Table 3 presents the performance of the B&B algorithm with the midpoint and monotonicity tests. The ranges of the functions and gradient enclosures are calculated using standard and min-range multiplication methods. The CPU time, the accuracy of computations ϵ, the number of repetitions of functions evaluation N, the number of iterations (#Itrs) and the number of elements remaining on the list at the end of the B&B algorithm (#Bxs) are compared.

Table 2 Test functions, search domains and optimal values

Function	Search domains and optima
$f_1(x) = (2x_1^2 - 1.05x_1^4 + x_2^2 - x_1x_2 - (1/6)x_2^6)$	$\boldsymbol{X} = [-5, 8]^2$ $x^* = (8, 8)$, $f_1(x^*) = -47863.467$
$f_2(x) = x_1^3x_2 + x_2^2x_3x_4^2 - 2x_5^2x_1 + 3x_2x_4^2x_5$	$\boldsymbol{X} = [-50, 50]^5$ $x^* = (-50, 50, 50, 50, -50)$, $f_2(x^*) = -337750000$
Booth function $f_3(x) = (x_1 + 2x_2 - 7)^2 + (2x_1 + x_2 - 5)^2$	$\boldsymbol{X} = [-10, 33.5] \times [-10, 34.5]$ $x^* = (1, 3),\ f_3(x^*) = 0$
Beale function $f_4(x) = (1.5 - x_1(1 - x_2))^2 + (2.25 - x_1 * (1 - x_2^2))^2$ $+ (2.625 - x_1 * (1 - x_2^3))^2$	$\boldsymbol{X} = [-5, 5]^2$ $x^* = (3, 0.5),\ f_4(x^*) = 0$
Golden & price function $f_5(x) = [1 + (x_1 + x_2 + 1)^2$ $\times (19 - 14x_1 + 13x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)]$ $\times [30 + (2x_1 - 3x_2)^2$ $\times (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$	$\boldsymbol{X} = [-4,\ 4]^2$ $x^* = (0, -1),\ f_5(x^*) = 3.0$
Himmelblau function $f_6(x) = (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2$	$\boldsymbol{X} = [0,\ 160]^2$ $x^* = (3, 2),\ f_6(x^*) = 0$
Ratschek (six hump) function $f_7(x) = 4x_1^2 - 2.1x_1^4 + (1/3)x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	$\boldsymbol{X} = [-2000, 2000]^2$ $x^* = (0.0898, -0.7126)$, $f_7(x^*) = -1.0316285$
Three hump function $f_8(x) = 12x_1^2 - 6.3x_1^4 + x_1^6 + 6x_2(x_2 - x_1)$	$\boldsymbol{X} = [-1000, 1000]^2$ $x^* = (0, 0),\ f_8(x^*) = 0$
Rosenbrock function $f_9(n, x) = \sum_{j=1}^{n} 100 * (x_j^2 - x_{j+1})^2 + (x_j - 1)^2$	$\boldsymbol{X} = [-5, 10]^n$ $x^* = (\underbrace{1, \ldots, 1}_{n}),\ f_9(x^*) = 0$
Dixon & price function $f_{10}(n, x) = (x_1 - 1)^2 + \sum_{i=2}^{n} i(2x_i^2 - x_{i-1})^2$	$\boldsymbol{X} = [-n, n]^n$

For all the considered functions B&B algorithm utilising the MR multiplication performs better. The MR method always gives the best number of iterations and also the smallest number of the boxes remaining in the list at the end of the program (except for f_5). Moreover, MR always gives the best CPU time. The gain of the average CPU time between ST and MR is about 30 %.

Therefore, in spite of that the min-range multiplication is computationally more complex, as it was shown in Example 1, it can be useful in practical applications.

Table 3 Global optimisation using standard and min-range multiplications: comparison of basic characteristics

Function	N	ϵ		#Itrs/#Bxs		CPU time [s]		CPU gain [%]
				ST	MR	ST	MR	
f_1	100	10^{-11}		6/5	3/2	0.156	0.062	60
f_2	100	10^{-12}		25/23	15/13	0.983	0.764	22
f_3	100	10^{-12}		304/52	270/45	5.944	5.413	9
f_4	100	10^{-12}		1738/155	1255/12	89.247	77.501	13
f_5	10	10^{-12}		2250/289	1644/352	28.424	23.899	16
f_6	100	10^{-12}		181/121	141/116	4.47	3.82	15
f_7	100	10^{-12}		714/218	554/160	42.81	25.44	41
f_8	100	10^{-12}		808/296	538/272	28.38	19.43	32
f_9	1	10^{-10}	$n=4$	90384/9739	81934/8744	50.267	49.546	1
			$n=5$	444933/53825	383454/46249	412.31	358.148	13
f_{10}	1	10^{-12}	$n=4$	20888/3833	18253/3563	10.426	9.898	5
			$n=5$	121142/25091	94241/20219	82.395	66.505	19
			$n=6$	706127/172254	505552/131661	998.408	583.93	42
Average				106885/19386	83681/16262	134.94	94.181	

Example 4 (Finding roots of rational functions) Consider the rational function with the third degree polynomial in the numerator and the second degree polynomial in the denominator:

$$f(x) = \frac{x^3 - 26x^2 + 209x - 492}{x^2 + 1}.$$

The aim is to find all roots of f in the interval $x_0 = [2, 15]$. Since the function is continuous and has continuous first derivative, thus the Interval Newton Method [1] can be used to find the roots. Interval extensions of f and df are computed using affine arithmetic. The range of the denominator computed with the use of trivial and standard multiplication equals $[x^2 + 1]_{\mathrm{TR}} = [-79.5,\ 226]$ and $[x^2 + 1]_{\mathrm{ST}} = [-37.25,\ 226]$, respectively. In both cases it contains zero, and therefore requires a special treatment (one can use extended division which yields a union of intervals or one can split the initial interval, perhaps more than once, and repeat evaluation of the function on subintervals; however, this requires extra computation time). The min-range multiplication gives the exact range $[5,\ 226]_{\mathrm{MR}}$, and thus the Interval Newton Method can be applied directly. The result is presented in Table 4.

All the roots were found in less than 0.0003 s.

Table 4 Interval Newton methods results for $f(x) = \frac{x^3-26x^2+209x-492}{x^2+1}$

Roots
$x_1 = [4.1715728752537933, 4.1715728752538279]$
$x_2 = [9.8284271247461934, 9.8284271247461952]$
$x_3 = [11.999999999999897, 12.000000000000034]$

5 Conclusions

A method for computing the min-range approximation of a product of affine forms was discussed. The new algorithm for computing extrema of the function $f(x,\ y) = xy$ on a joint range $\langle \hat{x},\ \hat{y} \rangle$ of two affine forms $\hat{x},\ \hat{y}$ was proposed. This algorithm considerably generalises Kolev's [7] multiplication method to arbitrary affine forms which not contain zero properly, since it computes the required extrema without monotonicity requirement.

Various numerical experiments were performed to validate the proposed algorithm. The obtained results show that despite the min-range multiplication is more time-consuming, it can be useful for solving optimisation problems or finding roots of functions. The proposed approach improves global optimisation efficiency by decreasing the number of iterations and consequently the overall computational time. The efficiency gains seem to grow along with the number of variables and the width of initial intervals grows.

The MR method can be useful as well when dealing with algebraic expressions involving square root, logarithmic or rational functions, since the narrower ranges are less likely to contain zero.

References

1. Alefeld, G.: Inclusion methods for systems of nonlinear equations—the interval Newton method and modifications. In: Herzberger, J. (ed.) Topics in Validated Computations. Elsevier, Amsterdam (1994)
2. Comba, J.L.D., Stolfi, J.: Affine arithmetic and its applications to computer graphics. In: Proc. SIBGRAPI'93–VI Simpósio Brasileiro de Computação Gráfica e Processamento de Imagens (Recife, BR), pp. 9–18 (1993)
3. de Figueiredo, L.H., Stolfi, J.: Adaptive enumeration of implicit surfaces with affine arithmetic. Comput. Graph. Forum **15**, 287–296 (1996)
4. de Figueiredo, L.H., Stolfi, J.: Self-validated numerical methods and applications. In: Brazilian Mathematics Colloquium Monograph, IMPA. Rio de Janeiro, Brazil (1997)
5. de Figueiredo, L.H., Stolfi, J.: An introduction to affine arithmetic. TEMA Tend. Mat. Apl. Comput. **4**(3), 297–312 (2003)
6. de Figueiredo, L.H., Stolfi, J.: Affine arithmetic: concepts and applications. Numer. Algorithms **37**(1–4), 147–158 (2004)
7. Kolev, L.V.: Optimal multiplication of G-intervals. Reliab. Comput. **13**, 399–408 (2007)
8. Kolev, L.V.: A new method for global solution of systems of nonlinear equations. Reliab. Comput. **4**(1), 1–21 (1998)
9. Zhang, Q., Martin, R.R.: Polynomial evaluation using affine arithmetic for curve drawing. In: Proc. Eurographics UK 2000 Conf., pp. 49–56. Eurographics UK, Abingdon (2000)
10. Martin, R.R., Shou, H., Voiculescu, I., Bowyer, A., Wang, G.: Comparison of interval methods for plotting algebraic curves. Comput. Aided Geom. Des. **19**(7), 553–587 (2002)
11. Moore, R.E.: Interval Analysis. Prentice-Hall, Englewood Cliffs (1966)
12. http://extreme.adorio-research.org/download/mvf/html/node3.html. Accessed August 2011
13. http://www.mat.univie.ac.at/ neum/glopt/moretest/. Accessed August 2011
14. http://www-optima.amp.i.kyoto-u.ac.jp/member/student/hedar/Hedar_files/TestGO_files/Page364.htm. Accessed August 2011